REMARKS

Claims 1-71 are pending, of which claims 1, 25, 48, 54, 55, 56, and 61, are independent. In the August 4, 2008 Response to the June 4, 2008 Office Action, Applicants canceled claim 10 without prejudice or disclaimer, and amended claimed 71 to correct a typographical problem. The claim listing beginning on page 2 of this Response reflects the claims as amended in the August 4, 2008 Response, with the following change: claims 11, 13, and 15 are amended herein to depend from claim 1. Previously, claims 11, 13, and 15 depended from claim 10, which was canceled in the August 4 Response.

Applicants respectfully request reconsideration of the outstanding rejections and passage of the claims to allowance.

Applicants thank the Examiner for withdrawing the rejection under 35 U.S.C. §112 in the June 4, 2008 Office Action. Applicants further thank the Examiner for withdrawing the objection to claim 71, and the 35 U.S.C. §112 rejections of claims 10, 70, and 71 in the August 13 Advisory Action (Advisory Action at page 2).

Applicants thank the Examiner for the courtesy of a telephonic conference on September 3, 2008, at which time the Advisory Action was discussed. The Examiner offered a number of helpful explanations and suggestions. Applicants have considered these explanations and suggestions when preparing this Response. The Examiner indicated that Applicants' arguments presented in the telephonic conference would overcome the specific rejections raised in the Advisory Action. Applicants respectfully include the relevant arguments in this Response.

I. Rejections under 35 U.S.C. §103(a)

A. Claims 1-3, 5-9, 17-22, and 24

In the Office Action, claims 1-3, 5-9, 17-22, and 24 were rejected under 35 U.S.C. §103(a) as being unpatentable over MathWorks Simulink® "Dynamic System Simulation for MATLAB," 1997 (hereinafter "MathWorks") in view of Official Notice taken. Applicants respectfully traverse the rejection.

Applicants' claim 1 recites,

1. A method comprising the steps of:

providing a graphical debugger interfaced with a model view of a model being executed, ***said model comprising a block that includes a plurality of execution methods***, said graphical debugger having debug information related to the execution of said model, ***said debug information indicating order of execution of said plurality of execution methods for said block and a start time or a stop time of said plurality of execution methods for said block that are executed during the execution of said model***; and

outputting said debug information to a user, ***said debug information allowing the user to determine proper or improper operation for at least a subset of said plurality of execution methods that are executed in said block during the execution of said model***.

MathWorks does not disclose or suggest ***said debug information indicating order of execution of said plurality of execution methods for said block and a start time or a stop time of said plurality of execution methods for said block that are executed during the execution of said model***, as present in claim 1.

In the August 13 Advisory Action, the Examiner asserts that "for said block" is a "possessive term, which means that the execution methods belong to the said block." (Advisory Action, page 2). Applicants agree with the Examiner that the term "for said block" indicates a relationship between the execution methods and the block. For example, as recited in the claim language, the block ***includes a plurality of execution methods***.

In the August 13 Advisory Action, the Examiner next asserts that the term "for said block" does not mean "that the order of execution is indicated such that the information contains the order of execution within the block." Applicants respectfully disagree. As indicated above, the Examiner correctly identifies that the execution methods are *related to the block*. As Applicants noted in the telephonic conference of September 3, although the model may comprise more than one block (the term "comprising" being open language), the plurality of execution methods are included in ***a block***. Thus, claim 1 recites indicating order of execution of a plurality of execution methods that are included in the single block. The Example which the Examiner described in the June 4 Office Action involves the *relative order* of execution methods included in *multiple blocks*. In that example, the *blocks* are debugged without presenting information regarding the order of execution of the execution methods included in ***a block***. In contrast, claim 1 recites ***said debug information indicating order of execution of said plurality***

of execution methods for said block.

The Examiner further asserts "it is believed that Applicants intend to claim (based on the arguments), indication the order of execution within said block" (Advisory Action, page 2). Applicants respectfully disagree with these characterizations of the claim language. In the telephonic conference of September 3, Applicants indicated that the Examiner's phrase "within the block" is not an accurate description of the relationship between the execution methods and the block, as recited in the actual claim language. For example, the claim language recites ***a block that includes a plurality of execution methods*** and ***said debug information indicating order of execution of said plurality of execution methods for said block***. The execution methods recited in claim 1 are not necessarily *executed* "within the block" as suggested by the Examiner.

In addition to the above assertions, the Examiner states (in the June 4 Office Action) that "given that MathWorks discloses the order of execution of blocks, and the blocks contain plurality of execution methods (sic), the order of the execution of methods is inherently provided" (Office Action at page 3, paragraph 6.1). The Examiner then provides the following example: "BLOCK1=E1,E2,E4,E5; BLOCK2 = E6,E7,E9. From this it is clearly seen that E1, E2, E4, and E5 are executed before E6, E7, E9. Therefore, an order of method execution has been established."

Applicants respectfully submit that the Examiner misreads claim 1. Claim 1 recites that the debug information indicates the ***order of execution of the plurality of execution methods for said block*** – the block referred to is ***a block*** in the model, as recited earlier in claim 1.

The Examiner appears to ignore the claim language reciting that the execution methods are ***for said block.*** Claim 1 describes obtaining the order, and start time or stop time, of execution methods included in a single block. This is not possible in the Examiner's example. In fact, the Examiner's example says nothing about debugging execution methods included in a block, as is required by claim 1. Applicants are not surprised by this outcome since MathWorks does not disclose or suggest debugging execution methods inside a block.

In determining the differences between the prior art and the claims, the question under 35 U.S.C. §103 is not whether the differences themselves would have been obvious, but whether the claimed invention as a whole would have been obvious (MPEP §2141.02(I), emphasis in

original). The Examiner appears to attack claim 1 piecemeal without giving any apparent consideration to the claim features as a whole. The analysis technique applied by the Examiner selectively and impermissibly excludes certain claim language (e.g., "for said block").

Referring to the Examiner's example, using MathWorks at best one could only determine that Block 1 executed before Block 2. Using MathWorks, one could not debug whether an execution error associated with Block 1 was caused by an error with one of the execution methods executing in Block 1, namely one of execution methods E1,E2,E4,E5.

Therefore, the order of execution of methods included in a single block is not inherently provided in the above example, as the Examiner suggests. Referring to the Examiner's example, even if a user knew that E1, E2, E4, and E5 are executed before E6, E7, and E9, this is not the same as ***debug information indicating order of execution of said plurality of execution methods for said block and a start time or a stop time of said plurality of execution methods for said block***. Claim 1 refers to the execution methods in a single block. The method of claim 1 would allow a user to learn that the methods of block 1 had executed in the order E1, E4, E2, E5, for example, or E2, E4, E5, E1. This is not possible using the method indicated by the Examiner.

Further, the order of execution of methods for a block is not necessarily constant. For example, the order of execution in a first simulation run might be E1, E2, E4, E5, but in a subsequent run, might be E4, E2, E5, E1. Even within the same simulation run, the order of execution may vary. Some types of blocks will call execution methods in a different order depending on what kind of input they receive. Thus, the Examiner's example does not represent ***debug information indicating order of execution of said plurality of execution methods for said block***.

Still further, the Examiner's example does not provide ***debug information indicating order of execution of said plurality of execution methods for said block and a start time or a stop time of said plurality of execution methods for said block that are executed during the execution of said model***. The method of claim 1 would allow a user to note (for example) that E1 began at 0:00 and ended at 0:05, E4 began at 0:06 and ended at 0:09, E2 began at 0:10 and ended at 0:12, and E5 began at 0:13 and ended at 0:17. The best that the example described by the Examiner could do is note that *some* method related to block 1 began at time 0:00, and *some*

method related to block 1 ended at 0:17. The identity of the particular methods would be unknown, because the Examiner's example cannot "look into" the block to determine the order of execution of the methods included in the block since MathWorks does not disclose looking into a block to debug particular execution methods executing in the block. This is not what is claimed. Claim 1 recites ***a start time or a stop time of said plurality of execution methods for said block***.

As the Applicants have consistently argued since the Interview of October 3, 2006, MathWorks is focused at a different degree of granularity than the present invention. The granularity of MathWorks is at the block level in a model; therefore, MathWorks cannot debug execution methods in a block. In contrast, claim 1 is directed to execution methods ***for*** a block.

For at least the reasons given above, MathWorks, even in view of the Official Notice taken, does not disclose or suggest each and every element of independent claim 1. Claims 2-3, 5-9, 17-22, and 24 depend from claim 1 and, as such, include each and every element of claim 1. Thus, MathWorks does not disclose or suggest each and every element of claims 2-3, 5-9, 17-22, and 24. Therefore, Applicants respectfully request that the 35 U.S.C. §103(a) rejection of claims 1-3, 5-9, 17-22, and 24 be withdrawn.

B. Claims 25-27, 29-33, 41-46, and 48-69

In the Office Action, claims 25-27, 29-33, 41-46, and 48-69 were rejected under 35 U.S.C. §103(a) as being unpatentable over MathWorks in view of Official Notice taken. Applicants respectfully traverse the rejection.

1. Claims 25-27, 29-33, and 41-46

Applicants' claim 25 recites,

> 25. A medium for use in a modeling and execution environment on an electronic device, said medium holding executable instructions on the electronic device for performing an execution method, said method comprising the steps of:
> providing a graphical debugger interfaced with a model view of a model being executed, said model comprising a block that includes a plurality of execution methods, said graphical debugger having debug information related to

the execution of said model, ***said debug information indicating at least one of the order of the execution of a plurality of execution methods in said model and a start time or a stop time of at least one execution method executed during the execution of said model***; and

outputting said debug information to a user, ***said debug information allowing the user to determine proper or improper operation for at least a subset of said plurality of execution methods for the block that are executed during the execution of said model.***

MathWorks does not disclose or suggest ***said debug information indicating at least one of the order of the execution of a plurality of execution methods in said model and a start time or a stop time of at least one execution method executed during the execution of said model***, nor does MathWorks disclose or suggest ***said debug information allowing the user to determine proper or improper operation for at least a subset of said plurality of execution methods for the block that are executed during the execution of said model***, as present in claim 25.

As described above with reference to claim 1, MathWorks provides a different level of granularity than claim 25. MathWorks focuses on blocks rather than execution methods, and does not disclose or suggest indicating the execution order of multiple execution methods included in a block, the start and stop time of multiple execution methods operating in a block, or allowing a user to determine proper or improper operation for multiple execution methods operating inside a block. Thus, MathWorks does not disclose or suggest all of the elements of claim 25.

The addition of the Official Notice does not cure the factual defects of MathWorks with respect to claim 25. As noted above, even if the blocks of MathWorks contain a plurality of execution methods, it is still not obvious to provide ***debug information indicating at least one of the order of the execution of a plurality of execution methods in said model and a start time or a stop time of at least one execution method executed during the execution of said model***, nor is it obvious that ***said debug information allows the user to determine proper or improper operation for at least a subset of said plurality of execution methods for the block that are executed during the execution of said model***, as recited by claim 25.

Claims 26-27, 29-33, and 41-46 depend from claim 25, and thus include each and every element of claim 25. In light of the above arguments, Applicants respectfully request that the 35 U.S.C. §103(a) rejection of claims 25-27, 29-33, and 41-46 be withdrawn.

2. Claims 48-53

Applicants' claim 48 recites,

> 48. A computer-implemented method, comprising:
> identifying a first execution method operating in a first environment of a computer-based modeling application that executes a model, where the first environment is one of a text-based environment, a time-based block diagram, a state based block diagram, or a data-flow diagram;
> identifying a second execution method operating in a second environment, where the second environment differs from the first environment;
> ***debugging the first execution method and the second execution method while the computer-based model operates on behalf of a user***; and
> ***generating output information for the user or for a destination, the output information identifying when the first execution method or the second execution method are operating, identifying an operation performed by the first execution method or the second execution method at a determined location in the first execution method or the second execution method, or identifying an error related to the first execution method or the second execution method during execution of the computer-based model***.

MathWorks does not disclose or suggest ***debugging the first execution method and the second execution method while the computer-based model operates on behalf of a user***, or ***generating output information for the user or for a destination, the output information identifying when the first execution method or the second execution method are operating, identifying an operation performed by the first execution method or the second execution method at a determined location in the first execution method or the second execution method, or identifying an error related to the first execution method or the second execution method during execution of the computer-based model***, as present in claim 48.

The Examiner states that "the methods are related to the blocks associated therewith. Therefore, the identification of errors within a block identifies the errors that are related to the methods associated therewith, which in this case are the execution methods." However, this statement does not describe debugging the execution methods, as recited in claim 48. As noted in claim 1, MathWorks is focused at a different degree of granularity than the present Application.

The Examiner asserts, in the August 13 Advisory Action, that "by indicating that an error occurs within a block having only a first and a second method, the indication inherently

identifies that the first or second method has the error." Applicants respectfully disagree with this assertion. It is not the case that indicating a block-level error will "inherently" identify that an execution method within the block "has the error." As the Applicants have noted (e.g., in the August 4 Response in relation to claim 61), a block may include other elements besides methods – blocks and methods are not "functionally equivalent," as the Examiner suggests. Even if a block contained only two methods, indicating that an error occurs within a *block* does not identify that the error has occurred in one of the methods included in the block.

Further, even if the Examiner were correct that indicating a block level error was somehow equivalent to ***identifying an error related to the first execution method or the second execution method during execution of the computer-based model***, MathWorks *still* would not disclose or suggest ***debugging the first execution method and the second execution method while the computer-based model operates on behalf of a user***, as recited in claim 48. MathWorks can identify, at best, block-level errors, and cannot debug the execution methods while the model operates.

Multiple execution methods may be present in a block, and knowing that a block has thrown an error will not tell a user which execution method internal to the block is the cause. Thus, MathWorks does not disclose or suggest ***identifying an error related to the first execution method or the second execution method during execution of the computer-based model***. An error identified in the Examiner's example on page 5 might belong to the first execution method, if the first execution method resides in a first block, but the error could just as easily be related to a third execution method, also associated with the first block. Thus, identifying errors on a block-level basis, as the Examiner suggests, is not equivalent to ***identifying an error related to the first execution method or the second execution method.*** Because MathWorks cannot "look into a block" to debug the block's execution methods (as noted above in relation to claim 1), MathWorks also does not disclose ***debugging the first execution method and the second execution method while the computer-based model operates on behalf of a user***.

Further, in the MathWorks debugger, a user can display a model's block execution order (MathWorks at page 12-16), but the user cannot identify when an execution method is operating. In the MathWorks debugger, users can set breakpoints at the beginning or end of a block, thus allowing them to identify when an operation is performed by a block (MathWorks at page 12-9), but users cannot identify an operation performed by an execution method at a determined

location inside the execution method. In MathWorks, users can step through the simulation block-by-block (MathWorks at page 12-5) and thus identify block-level errors; however, MathWorks does not allow a user to identify an error related to the first execution method or the second execution method during execution of the computer-based model. In fact, MathWorks is silent as to these features. Thus, MathWorks does not disclose or suggest all the elements of claim 48.

The Official Notice taken fails to cure the factual deficiencies of MathWorks with regard to claim 48. As noted above in relation to claim 1, even if the blocks of MathWorks do contain a plurality of execution methods, it is still not obvious to debug those execution methods. Therefore, MathWorks, even in view of the official notice taken, does not disclose or suggest at least ***debugging the first execution method and the second execution method while the computer-based model operates on behalf of a user***, nor is it obvious to ***generating output information for the user or for a destination, the output information identifying when the first execution method or the second execution method are operating, identifying an operation performed by the first execution method or the second execution method at a determined location in the first execution method or the second execution method, or identifying an error related to the first execution method or the second execution method during execution of the computer-based mode***, as recited in claim 48.

Claims 49-53 depend from claim 48, and thus include each and every element of claim 48. In light of the above arguments, Applicants respectfully request that the 35 U.S.C. §103(a) rejection of claims 48-53 be withdrawn.

3. Claim 54

Claim 54 recites,

> 54. A method, comprising:
> receiving information about a first execution method and a second execution method on behalf of a graphical model comprising blocks, where at least one of the blocks includes the first execution method and at least one other execution method or the second execution method and the at least one other execution method, where the first execution method or the second execution method are related to one or more of the blocks;
> ***identifying at least a portion of the first execution method or the second execution method when the first execution method or the second execution***

method are running, respectively;

obtaining information about the running of the first execution method or the second execution method using the identifying; and

providing debugging information to a user via a display or providing debugging information to a destination device, ***the debugging information identifying the first execution method or the second execution method and information about the first execution method or the second execution method, respectively***.

MathWorks does not disclose or ***suggest identifying at least a portion of the first execution method or the second execution method when the first execution method or the second execution method are running, respectively***, nor does MathWorks disclose or suggest ***obtaining information about the running of the first execution method or the second execution method using the identifying***, nor does MathWorks disclose or suggest ***the debugging information identifying the first execution method or the second execution method and information about the first execution method or the second execution method, respectively***, as present in claim 54.

The Examiner provides no further support for the rejection of claim 54 beyond the support offered for the rejection of claim 48 (Office Action at page 5). This is improper, as claim 54 recites elements not found in claim 48. For example, claim 54 recites ***obtaining information about the running of the first execution method or the second execution method using the identifying***.

Even if the rejection were proper, MathWorks, even in view of the Official Notice taken, does not disclose or suggest each and every element of claim 54. MathWorks does not identify a portion of an execution method when the execution method is running. Instead, MathWorks identifies which blocks are being executed (MathWorks at page 12-16). MathWorks does not obtain information about the running of first execution methods using that identification, but rather obtains information on overall system states and individual block states (MathWorks at pages 12-12 to 12-14). And MathWorks does not provide debugging information identifying execution methods, but rather debugging information pertaining to block execution (MathWorks at page 12-16).

Therefore, MathWorks, even in view of the Official Notice taken, does not disclose each and every element of claim 54. Applicants respectfully request that the 35 U.S.C. §103(a)

rejection of claim 54 be withdrawn.

4. Claim 55

Claim 55 recites,

55. A method, comprising:
identifying a first root method comprising one or more child methods, the first root method related to a graphical modeling application;
identifying a second root method related to the graphical modeling application;
running the first root method and the second root method in a graphical debugger to obtain information about the operation of the first root method or the second root method; and
displaying a debugging result to a destination, ***the debugging result comprising visual identifiers related to the operation of the first root method, the one or more child methods or the second root method, error information about the first root method, the one or more child methods or the second root method, an execution result for the first root method, the one or more child methods or the second root method, or status information related to the first root method, the one or more child methods or the second root method***.

MathWorks does not disclose or ***the debugging result comprising visual identifiers related to the operation of the first root method, the one or more child methods or the second root method, error information about the first root method, the one or more child methods or the second root method, an execution result for the first root method, the one or more child methods or the second root method, or status information related to the first root method, the one or more child methods or the second root method***, as present in claim 55.

The Examiner asserts (in relation to claim 61) that blocks and methods are "functionally equivalent," and therefore, "because the reference discloses the root-child (hierarchic nature) at a block level, it inherently does so at the level that is beneath it, the method-level" (Office Action at page 6). Applicants respectfully disagree with the Examiner's contention. There is nothing in either MathWorks or the present Application that states, implicitly or explicitly, that blocks are "merely a group of methods." Applicants note, in the Application at page 3, that blocks "***contain*** a collection of methods that are invoked by the execution engine at certain times during the simulation for different purposes." While blocks may "contain" a collection of methods, it is not accurate to suggest that blocks and methods are "functionally equivalent." For example, a block

may contain, in addition to a collection of methods, data, state information, or an iconic representation (see, e.g., Application at page 14-15).

On page 3 of the August 13 Advisory Action, the Examiner asserts that "Applicants are arguing features which are not necessitated by the claims. Specifically, the addition of collection of methods (sic), data, state information, or an iconic representation is not claimed." Applicants respectfully submit that they are not "arguing features which are not necessitated by the claims," but rather traversing the Examiner's supporting rationale. Claim 55 *does not recite a block at all*, but rather recites a first and second ***root method related to a graphical modeling application***. While "a collection of methods, data, state information, or an iconic representation" are not recited in the claim, they are examples of why blocks are not merely "a collection of methods," as the Examiner asserts in the Office Action and Advisory Action. The Examiner asserts that a block is the same as a root method; however, this is incorrect for at least the reasons given above. It is not necessary to claim a block in order to acknowledge that a block is not "functionally equivalent" to a method.

The Examiner asserts that, because blocks and methods are "functionally equivalent," and, "because the reference discloses the root-child (hierarchic nature) at a block level, it inherently does so at the level that is beneath it, the method-level" (Office Action at page 6). As noted, blocks and methods are not "functionally equivalent," as the Examiner suggests. It is therefore not accurate to say that the reference "inherently" discloses a root-child relationship at the method-level.

MathWorks discusses blocks and not root methods. Thus, MathWorks does not give a debugging result with visual identifiers related to the operation of root or child methods, but instead gives visual identifiers related to the execution of blocks (MathWorks at page 12-16). MathWorks does not disclose or suggest displaying status information related to root or child methods, but can display status information related to blocks and overall system states (MathWorks at pages 12-12, 12-14, and 12-16).

Thus, MathWorks, even in view of the Official Notice taken, does not disclose or suggest all the elements of claim 55. Therefore, Applicants respectfully request that the 35 U.S.C. §103(a) rejection of claim 55 be withdrawn.

5. Claims 56-60

Claim 56 recites,

56. A method for implementing a user interface for debugging a graphical model, the method comprising:

displaying a hierarchy comprising information about a first root method, one or more child methods related to the first root method, or a second root method, the hierarchy displaying information about the first root method, the one or more child methods, or the second root method in an arrangement representing a relationship among the first root method, the one or more child methods, or the second root method; and

displaying an indicator on the hierarchy proximate to the first root method, the one or more child methods, or the second root method, the indicator denoting a status of the first root method, the one or more child methods, or the second root method, where the status indicates whether the first root method, the one or more child methods, or the second root method are operating according to determined parameters.

MathWorks does not disclose or suggest ***displaying a hierarchy comprising information about a first root method, one or more child methods related to the first root method, or a second root method, the hierarchy displaying information about the first root method, the one or more child methods, or the second root method in an arrangement representing a relationship among the first root method, the one or more child methods, or the second root method***, nor does MathWorks disclose or suggest ***displaying an indicator on the hierarchy proximate to the first root method, the one or more child methods, or the second root method, the indicator denoting a status of the first root method, the one or more child methods, or the second root method, where the status indicates whether the first root method, the one or more child methods, or the second root method are operating according to determined parameters***, as present in claim 56.

The Examiner asserts that blocks and methods are "functionally equivalent," and therefore, "because the reference discloses the root-child (hierarchic nature) at a block level, it inherently does so at the level that is beneath it, the method-level" (Office Action at page 6). As noted above, blocks and methods are not "functionally equivalent," as the Examiner suggests. It is therefore not accurate to say that the reference "inherently" discloses a root-child relationship at the method-level.

Instead of displaying a hierarchy of root and child methods, MathWorks displays a block

execution order. For instance, compare MathWorks at page 12-16, showing the hierarchy of MathWorks, with Figure 18A of the Application, which depicts the parent-child hierarchy. The block execution order of MathWorks is silent as to with root or child methods.

Thus, MathWorks, even in view of the Official Notice taken, does not disclose or suggest all the elements of claim 56. Claims 57-60 depend from claim 56, and thus include ever element of claim 56. In light of the above arguments, Applicants respectfully request that the 35 U.S.C. §103(a) rejection of claims 56-60 be withdrawn.

6. Claims 61-69

Claim 61 recites,

> 61. A method for debugging operation of a graphical icon, the method comprising:
> ***identifying a plurality of execution methods for the graphical icon using a plurality of regions related to the graphical icon***;
> ***displaying information about a first one of the plurality of execution methods in a first one of the plurality of regions or information about a second one of the plurality of execution methods in a second one of the plurality of regions***; and
> associating the information in the first one of the plurality of regions or information in the second one of the plurality of regions with a graphical debugger to provide a user with debugging results for the first one of the plurality of execution methods or the second one of the plurality of execution methods, the debugging results allowing the user to identify desirable operations performed on behalf of the graphical icon or undesirable operations performed on behalf of the graphical icon.

MathWorks does not disclose or suggest ***identifying a plurality of execution methods for the graphical icon using a plurality of regions related to the graphical icon,*** nor does MathWorks disclose or suggest ***displaying information about a first one of the plurality of execution methods in a first one of the plurality of regions or information about a second one of the plurality of execution methods in a second one of the plurality of regions***, as present in claim 61.

The Examiner asserts that blocks and methods are "functionally equivalent," and therefore, "because the reference discloses the root-child (hierarchic nature) at a block level, it inherently does so at the level that is beneath it, the method-level" (Office Action at page 6). As

noted above, blocks and methods are not "functionally equivalent," as the Examiner suggests. It is therefore not accurate to say that the reference "inherently" discloses a root-child relationship at the method-level.

MathWorks does not display information about execution methods. MathWorks obtains information only on individual block states and overall system states (MathWorks at pages 12-12 to 12-14). Thus, MathWorks does not identify execution methods for a graphical icon, as present in claim 61. MathWorks is also silent as to displaying information about execution methods in a plurality of regions, as present in claim 61.

Thus, MathWorks, even in view of the Official Notice taken, does not disclose or suggest all the elements of claim 61. Claims 62-69 depend from claim 61, and thus include ever element of claim 61. In light of the above arguments, Applicants respectfully request that the 35 U.S.C. §103(a) rejection of claims 61-69 be withdrawn.

C. Claims 4, 10-16, 23, 28, 34-40, and 47

Claims 4, 10-16, 23, 28, 34-40, and 47 were rejected under 35 U.S.C. §103(a) as being unpatentable over MathWorks Simulink® (1997) ("MathWorks") as applied to claim 1 above, and further in view of Fenlason's "GNU gprof" (1998). Applicants respectfully traverse the rejection.

Claims 4, 10-16, 23, 28, 34-40, and 47 depend from claim 1 and, as such, include each and every element of claim 1. The Examiner adds GNU gprof to MathWorks to provide the specific limitations recited by these dependent claims, but does not assert that GNU gprof discloses the features of claims 1 and 25, from which they depend. For example, GNU gprof does not disclose or suggest features of claim 1, such as ***providing a graphical debugger interfaced with a model view of a model being executed, said model comprising a block***, nor does GNU gprof disclose or suggest ***said debug information indicating order of execution of said plurality of execution methods for said block and a start time or a stop time of said plurality of execution methods that are executed during the execution of said model.***

GNU gprof is a profiler used to determine which parts of a program are taking the most execution time (GNU gprof at 1). GNU gprof is used to profile programs, not models including blocks. Thus, GNU gprof does not disclose or suggest a debugger interfaced with a model view

of a model being executed, said model comprising ***a block***, nor does GNU gprof provide debug information indicating the order of execution of execution methods ***for blocks***. Because GNU gprof does not disclose execution methods in blocks, a feature also missing from MathWorks, GNU gprof does not remedy the shortcomings of MathWorks with respect to at least the above-mentioned features of claim 1.

The Examiner asserts that GNU gprof "disclsoes (sic) a model comprising blocks having a plurality of execution methods" because GNU gprof "disclose s'Compiling with ... augments your program with basic-block counting code'" (sic) (Office Action at page 6). In computer science, a basic-block is code that has one entry point, one exit point and no jump instructions contained within it. It is not related in any way to ***a model view of a model being executed, said model comprising a block***, as present in claim 1.

For at least the reasons presented above, MathWorks, the Official Notice taken, and GNU gprof, alone or in any reasonable combination, do not disclose each and every element of independent claim 1. Claims 4, 10-16, 23, 28, 34-40, and 47 depend from claim 1 and, as such, include each and every element of claim 1. Therefore, MathWorks, the Official Notice taken, and GNU gprof do not disclose each and every element of claims 4, 10-16, 23, 28, 34-40, and 47. Applicants therefore respectfully request that the 35 U.S.C. §103(a) rejection of claims 4, 10-16, 23, 28, 34-40, and 47 be withdrawn.

CONCLUSION

Applicants believe the pending application is in condition for allowance. If the Examiner feels that further discussion would expedite the proceedings, the Examiner is urged to call Applicants' attorney at the phone number listed below.

Please charge any shortage or credit any overpayment of fees to our Deposit Account No. 12-0080, under Order No. MWS-106RCE. In the event that a petition for an extension of time is required to be submitted herewith, and the requisite petition does not accompany this response, the undersigned hereby petitions under 37 C.F.R. §1.136(a) for an extension of time for as many months as are required to render this submission timely. Any fee due is authorized to be charged to the aforementioned Deposit Account.

Dated: September 4, 2008

Respectfully submitted,

Electronic signature: /Kevin J. Canning/
Kevin J. Canning
Registration No.: 35,470
LAHIVE & COCKFIELD, LLP
One Post Office Square
Boston, Massachusetts 02109-2127
(617) 227-7400
(617) 742-4214 (Fax)
Attorney/Agent For Applicant